Adding Additional Light Sources

1. Add three command buttons.

2. Set the command button captions to
    a. "Create Dir Light".
    b. "Create Point Light".
    c. "Create Spot LIght".

3. Edit the first command button and enter the code as follows:

```
Private Sub Command2_Click()
        Inv3d1.createDirectLight ("Direct_Light")
End Sub
```

4. Edit the second command button and enter the code as follows:

```
Private Sub Command3_Click()
        Inv3d1.createPointLight ("Point_Light")
End Sub
```

5. Edit the third command button and enter the code as follows:

```
Private Sub Command4_Click()
          Inv3d1.createSpotLight ("Spot_Light")
End Sub
```

6. Run the application.

7. Open a file.

8. Click on "Create Dir Light", "Create Point Light" or "Create Spot Light". A light icon is inserted into the scene.

Alphabetical Reference

getLightType
getNextCamera
getNextLight
getNumCameras
getNumSelected
getSeekTime
getStillDrawStyle
getViewerType
hideAllLights
importSceneGraph
importSceneGraphEx
ImportFile
isAutoClipping
isDecorating
isDetailSeek
isFeedbackVisible
isHeadlight
isLightOn
isPopupMenuEnabled
isSelectionEnable
isUrlPickEnable
isViewing
isValidFile
isValidFileEx
isWWWEnable
OpenFile
pointCameraAt
readSceneGraph
readSceneGraphEx
resetToHomePosition
SaveFile
saveHomePosition
saveSceneGraph
SelectViewer
serializeControl
serializeSceneGraphEx
setAntialiasing
setAutoClipping
setBufferingType
setCameraAspectRatio
setCameraFarDistance
setCameraHeight
setCameraName
setCameraNearDistance
setCameraOrientation
setCameraPosition

setCameraScaleHeight
setCenterballManip
setCurrentCamera
setDetailSeek
setExaminerViewer
setFeedbackSize
setFeedbackVisibility
setFlyViewer
setFog
setHandleboxManip
setHeadlight
setInteractiveDrawStyle
setJackManip
setLightColor
setLightDirection
setLightDropOffRate
setLightIntensity
setLightLocation
setLightName
setNoneManip
setPlaneViewer
setPopupMenuEnabled
setReplaceAllManip
setSeekTime
setSelectionEnable
setSelectionMethod
setStillDrawStyle
setTabboxManip
setTrackballManip
setTransformBoxManip
setTransparencyType
setUrlPickEnable
setViewerType
setViewing
setWalkViewer
setWWWEnable
showAllLights
stopAnimating
turnOffLight
turnOnLight
validateCameraId
validateLightId
viewAll
viewSelection

BringWindowToTop

Bring the applications main window to the top of the Z order.

void BringWindowToTop(**void**)

Remarks
This event is fired when the Visual 3Space control has finished loading a file after being requested to do so via the remote control interface.

See Also
wwwEnable

Camera Management

copyCamera
createPerspectiveCamera
deleteCamera
EditCameras
getCameraAspectRatio
getCameraFarDistance
getCameraFocalDistance
getCameraHeight
getCameraName
getCameraNearDistance
getCameraOrientation
getCameraPosition
getCameraType
getCurrCameraId
getFirstCamera
getNextCamera
getNumCameras
pointCameraAt
resetToHomePosition
saveHomePosition
setCurrentCamera
setCameraAspectRatio
setCameraFarDistance
setCameraHeight
setCameraName
setCameraNearDistance
setCameraOrientation
setCameraPosition
setCameraScaleHeight
setCurrentCamera
validateCameraId

Category Reference

Contents

Conventions

Methods within the Visual 3Space Control that correspond directly to methods in the Open Inventor viewer classes retain the name of the Open Inventor method. In the Open Inventor naming convention, method names begin with lowercase. The Visual 3Space Control adheres to the additional convention that all method names beginning with lowercase **do not** display dialog boxes. Methods that display dialog boxes requiring user input always begin with uppercase.

EditAmbientLight

Displays a modeless dialog box that allows the ambient light color to be edited.

(void) editAmbientLight(void)

Parameters

none

Return Values

none

Remarks

Displays the Ambient Light editing dialog.

See Also

setLightColor
getLightColor

EditBackgroundColor

Displays a modeless dialog box that allows the background color to be edited.

void EditBackgroundColor(**void**)

Return Value

None.

Remarks

The background color editor dialog box displays a color wheel that allows simple selection of a background color. In addition, RGB, HSV, RGBV, and RGBHSV color editing paradigms are supported.

EditCameras

Displays the camera editor modal dialog box.

void EditCameras(**void**)

Return Value

None.

Remarks

The camera editor allows interactive manipulation of the Visual 3Space Control's cameras.

See Also

createPerspectiveCamera

EditDxfProperties

Display modal dialog containing DXF import properties.

void EditDxfProperties()

Return Value

None.

Parameters

None.

Remarks
Call this method to display the DXF property page in a modal dialog box.

See Also
dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

EditHeadlight

Displays a modeless dialog box that allows the headlight direction and color to be edited.

void EditHeadlight()

Return Value

None.

Parameters

None.

Remarks

The headlight editor is a modeless dialog box that enables the manipulation of the direction and color of the headlight.

EditLightColor

Displays a modeless dialog box that allows the color of the target light to be edited.

void EditLightColor(long *lLightId*)

Return Value

None.

Parameters

lLightId The id of the target light to edit.

Remarks

The color editor dialog box displays a color wheel that allows simple selection of a color. In addition, RGB, HSV, RGBV, and RGBHSV color editing paradigms are supported.

See Also
setLightColor
getLightColor

EditLights

Displays the light editor modal dialog box.

void EditLights(**void**)

Return Value

None.

Remarks

The light editor allows interactive manipulation of the Visual 3Space Control's lights.

See Also

createDirectLight
createPointLight
createSpotLight

EditSelectionColor

Displays a modeless dialog box that allows the color of the current selection to be edited.

void EditSelectionColor(**void**)

Return Value

None.

Remarks

The color editor dialog box displays a color wheel that allows simple selection of a color. In addition, RGB, HSV, RGBV, and RGBHSV color editing paradigms are supported.

See Also
EditSelectionMaterial

EditSelectionMaterial

Displays a modeless dialog box that allows the material properties of the current selection to be edited.

void EditSelectionMaterial(void)

Return Value

None.

Remarks

The material editor dialog box provides controls that allow the current selection's ambient, diffuse, and specular reflection coefficients to be interactively modified. In addition the emissive properties of the selection can be modified. The appearance of the current selection is updated interactively as the properties are changed.

See Also

EditSelectionColor

Event Reference

BringWindowToTop
UpdatePickUrlName
UpdateStatusText
UpdateUrlName
UrlPicked

File Manipulation

ImportFile
importSceneGraph
OpenFile
readSceneGraph
SaveFile
saveSceneGraph

ImportFile

Import an Open Inventor or VRML file.

boolean ImportFile(**void**)

Return Value

If successful ImportFile returns TRUE. Otherwise, it returns FALSE.

Remarks

ImportFile displays the FileOpen common dialog to allow selection of a file for import. The imported file is placed into the scene as the right most node in the scene graph. Open Inventor and VRML files are supported.

See Also

importSceneGraph
OpenFile
readSceneGraph
SaveFile
saveSceneGraph
isValidFile
isValidFileEx

Light Source Management

createDirectLight
createPointLight
createSpotLight
deleteLight
EditAmbientLight
EditLightColor
getFirstLight
getLightColor
getLightDirection
getLightDropOffRate
getLightId
getLightIntensity
getLightLocation
getLightName
getLightType
getNextLight
hideAllLights
isHeadlight
isLightOn
setHeadlight
setLightColor
setLightDirection
setLightDropOffRate
setLightIntensity
setLightLocation
setLightName
showAllLights
turnOffLight
turnOnLight
validateLightId

LoadFile

Method Reference

OpenFile

Open an Open Inventor or VRML file.

boolean OpenFile(**void**)

Return Value

If successful OpenFile returns TRUE. Otherwise, it returns FALSE.

Remarks

OpenFile displays the FileOpen common dialog to allow selection of a file to be opened. The current scene graph is discarded and replaced with the contents of the selected file. Open Inventor and VRML files are supported.

See Also

ImportFile
importSceneGraph
readSceneGraph
SaveFile
saveSceneGraph
isValidFile
isValidFileEx

Overview

The Visual 3Space(TM) Browser Control

The Visual 3Space Control from TGS is a complete 3D browser component that supports VRML 1.0 and Open Inventor(TM) 2.x formats. It delivers a robust and flexible browser object to Visual C++ and Visual Basic developers who wish to quickly add interactive 3D graphics to their applications.

The Visual 3Space Control is perfect for addressing application requirements in 3D user interface, CAD groupware, VRML browser, data visualization and simulation markets. As a browser and viewer object, the Visual 3Space Control can accept data created from PTC Pro/Engineer, SDRC Ideas, Adobe 3DStudio and other applications supporting either VRML or Open Inventor formats. It is also possible to generate your own data files for the Visual 3Space Control using the Open Inventor SDK from Silcon Graphics or Template Graphics Software.

The Visual 3Space Control is ideal for local or internet-based applications. It supports the DDE interface used by Netscape and Microsoft, and can be used in a number of popular OCX container applications, such as Microsoft Word, Excel, Access and PowerPoint(TM).

The Visual 3Space Control was developed using the Open Inventor for Win32 SDK from TGS, and was modeled after the popular SceneViewer application. Over 150 methods and a robust property set enable you to design applications for manipulating 3D scenes. If you need additional functionality not contained in the Visual 3Space Control, contact TGS for information about the Open Inventor for Win32 SDK.

Property Reference

text3dRed
text3dGreen
text3dBlue
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize
transparencyType
urlPickEnable
viewingOn
viewerType
wwwEnable
wwwDdeServiceName

SaveFile

Save the scene to an Open Inventor file.

void SaveSceneGraphToFile(**void**)

Return Value

None.

Remarks

Displays the FileSave common dialog to allow the name and path to be entered.

See Also

ImportFile
importSceneGraph
OpenFile
readSceneGraph
saveSceneGraph

SelectViewer

Displays a dialog that allows selection of Examiner, Walk, or Fly viewers.

void SelectViewer(**void**)

Return Value

None.

Remarks

The active viewer in the Visual 3Space Control is changed to the viewer type selected.

Selection Management

EditSelectionColor
EditSelectionMaterial
setCenterballManip
setHandleboxManip
setJackManip
setNoneManip
setReplaceAllManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setSelectionMethod

UpdatePickUrlName

Update the text that displays the URL of the geometry currently under the pointer.

void UpdatePickUrlName(**BSTR *** *lpbstrUrlName*)

Remarks

This event is fired by the Visual 3Space control when the mouse pointer is moved onto geometry that contains a URL name, or when the mouse pointer is moved off of a piece of geometry that contains a URL name. In the latter case, the string "" is passed.

See Also

wwwEnable
UrlPicked
UpdateUrlName

UpdateStatusText

Update status bar text.

void UpdateStatusText(**BSTR *** *lpbstrStatusText*)

Remarks
This event is fired by the Visual 3Space control during processing of requests from the remote control interface.

See Also
wwwEnable

UpdateUrlName

Update the displayed URL (location).

void UpdateUrlName(**BSTR *** *lpbstrUrlName*)

Remarks

This event is fired by the Visual 3Space control after a URL has been successfully loaded.

See Also

wwwEnable
UpdatePickUrlName
UrlPicked
urlPickEnable

UrlPicked

Event fired when the left mouse button is clicked while the pointer is over a WWW anchor.

void UpdatePickUrlName(**BSTR *** *lpbstrUrlName*)

Remarks

This event is fired by the Visual 3Space control when the mouse pointer is moved onto geometry that contains a URL name (a WWW Anchor) and the left mouse button is clicked. The URL name is supplied by the parameter lpbstrUrlName.

See Also

wwwEnable
UpdatePickUrlName
UpdateUrlName
urlPickEnable

Viewer Attributes

EditBackgroundColor
getBufferingType
getFeedbackSize
getInteractiveDrawStyle
getSeekTime
getStillDrawStyle
getViewerType
isAutoClipping
isDecorating
isDetailSeek
isFeedbackVisible
isHeadlight
isPopupMenuEnabled
isViewing
resetToHomePosition
saveHomePosition
setAntialiasing
setAutoClipping
setBufferingType
setDetailSeek
setFeedbackSize
setFeedbackVisibility
setFog
setHeadlight
setInteractiveDrawStyle
setSeekTime
setStillDrawStyle
setTransparencyType
setViewing
stopAnimating
setTransparencyType
setViewing
viewAll

Viewer Selection

SelectViewer
setExaminerViewer
setFlyViewer
setPlaneViewer
setWalkViewer

addViewPoint

Add the values of the specified camera to the view point list.

void addViewPoint(**long** *lCameraId,* **long** *lDuration)*

Return Value

None

Parameters

lCameraId	Source camera ID
lDuration	The time, in milliseconds, to view the scene before switching to the next view point.

Remarks

Add the values of the specified camera to the view point list.

See Also

firstViewPoint
lastViewPoint
nextViewPoint
priorViewPoint
resetViewPoints
traverseViewPoints

antiAliasing

Enable/Disable antialiasing.

Possible Values

TRUE or FALSE.

Remarks

Antialiasing techniques reduce jagged lines and make the objects drawn on the screen appear smooth.

See Also

setAntialiasing

autoClipping

Turns autoclipping on of off.

Possible Values

TRUE	turn on autoclipping.
FALSE	turn off autoclipping.

Remarks

When on, the near and far camera clipping planes are continuously adjusted around the scene's bounding box to minimize clipping. Autoclipping is on by default.

See Also

isAutoClipping
setAutoClipping

bufferingType

The buffering type of the current viewer.

Possible values are:

0 Single Buffered
1 Double Buffered
2 Interactive

Remarks

In single buffered mode, the image is rendered directly to the window. This results in flickering between redraws but uses less memory. In double buffered mode, the image is rendered to a back buffer and copied to the viewing window. This reduces flicker. Interactive buffer mode uses double buffering when the user is doing interactive work and single buffering otherwise.

See Also

setBufferingType
getBufferingType

copyCamera

Copies camera parameters from one camera to another.

boolean copyCamera(**long** *nDstId,* **long** *nSrcId)*

Return Value

If successful, copyCamera returns TRUE. Otherwise, it returns FALSE.

Parameters

nSrcId Source camera ID

nDstId Destination camera ID

Remarks

The camera parameters of camera nDstId are replaced with those contained in camera nSrcId.

See Also

createPerspectiveCamera
getFirstCamera
getNextCamera

createDirectLight

The **createDirectLight** method creates a directional light and inserts it into the scene.

```
long createDirectLight(
        LPCTSTR lpszLightName           //user defined light name
        );
```

Parameters

lpszLightName
: A user defined light name.

Return Values

If the method succeeds, the return value is a light id greater than or equal to zero.

If the method fails, the return value is -1.

Remarks

A directional light is created and inserted into the scene.

See Also

createPointLight
createSpotLight

createPerspectiveCamera

The **createPerspective** method creates a perspective camera.

```
long createPerspectiveCamera(
        LPCTSTR lpszCamName          //user defined camera name
        );
```

Parameters

lpszCamName
: A user defined camera name.

Return Values

If the method succeeds, the return value is a camera id greater than or equal to zero.

If the method fails, the return value is -1.

See Also

setCurrentCamera
deleteCamera

createPointLight

The **createPointLight** method creates a point light and inserts it into the scene.

```
long createPointLight(
        LPCTSTR lpszLightName           //user defined light name
        );
```

Parameters

lpszLightName
: A user defined light name.

Return Values

If the method succeeds, the return value is a light id greater than or equal to zero.

If the method fails, the return value is -1.

Remarks

A point light is created and inserted into the scene.

See Also

createDirectLight
createSpotLight

createSpotLight

The **createSpotLight** method creates a apot light and inserts it into the scene.

```
long createSpotLight(
        LPCTSTR lpszLightName           //user defined light name
        );
```

Parameters

lpszLightName
: A user defined light name.

Return Values

If the method succeeds, the return value is a light id greater than or equal to zero.

If the method fails, the return value is -1.

Remarks

A spot light is created and inserted into the scene.

See Also

createDirectLight
createPointLight

decorationOn

Turn the decorations on or off.

Possible Values

True	The fly wheels are visible.
False	The fly wheels are not visible.

See Also

isDecorating

deleteCamera

Deletes a camera.

boolean deleteCamera(**long** *nCameraId)*

Return Value

If successful, deleteCamera returns TRUE. Otherwise, it returns FALSE.

Parameters

nCameraId ID of camera to be deleted

Remarks

The camera is destroyed and it's associated resources are released.

See Also

createPerspectiveCamera

deleteLight

Deletes a light.

void deleteLight(**long** *lLightId)*

Return Value

None.

Parameters

lLightd ID of light to be deleted

Remarks

The light is destroyed and it's associated resources are released.

See Also

createDirectLight
createPointLight
createSpotLight

deleteSceneGraph

Deletes the current scene graph.

void deleteSceneGraph(*)*

Parameters

None.

Remarks

The scene graph is destroyed when deleteSceneGraph is called. This will cause any objects displayed by the active viewer to dissappear. Note that dropping VRML or Open Inventor files on the viewer window will delete the current scene graph and load the dropped file.

See Also

importSceneGraph
readSceneGraph
saveSceneGraph

detailSeekOn

Turns detail seek on of off.

Possible Values

TRUE	turn on detail seek (Default).
FALSE	turn off detail seek.

Remarks

When the viewer is in seek mode, left mouse clicks initiate a pick, and the viewer changes its orientation and position to look at the picked object. The detailSeekOn property tells the seeking viewer whether to orient the camera towards the picked point (detailSeekOn TRUE), or the center of the object's bounding box (detailSeekOn FALSE).

See Also

isDetailSeek
setDetailSeek

dragDropEnable

Enable/Disable drag and drop.

Possible Values

TRUE or FALSE.

Remarks

Set dragDropEnable to TRUE or FALSE to enable or disable drag and drop respectively. The default value of the property is TRUE.

dxfAmbientScaleB

DXF import ambient blue scale factor.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffusse color only. The ambient color for each imported entity is derived by scaling the entity diffuse color by the dxfAmbientScale[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfAmbientScaleG

DXF import ambient green scale factor.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffusse color only. The ambient color for each imported entity is derived by scaling the entity diffuse color by the dxfAmbientScale[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfAmbientScaleR

DXF import ambient red scale factor.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffusse color only. The ambient color for each imported entity is derived by scaling the entity diffuse color by the dxfAmbientScale[RGB] properties.

See Also

dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfCreaseAngle

DXF import crease angle.

Possible Values

Floating point value specifying an angle in radians.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. Open Inventor calculates normals for imported DXF entities and the crease angle affects this computation. The crease angle indicates the minimum angle (in radians) between two adjacent face normals required to form a sharp crease at the edge.

See Also

dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType

dxfEmissiveScaleB

DXF import emissive blue scale factor.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffusse color only. The emissive color for each imported entity is derived by scaling the entity diffuse color by the dxfEmissiveScale[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfEmissiveScaleG

DXF import emissive green scale factor.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffusse color only. The emissive color for each imported entity is derived by scaling the entity diffuse color by the dxfEmissiveScale[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfEmissiveScaleR

DXF import emissive red scale factor.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffusse color only. The emissive color for each imported entity is derived by scaling the entity diffuse color by the dxfEmissiveScale[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfFaceType

DXF import face type.

Possible Values

```
typedef enum {
        UNKNOWN_FACE_TYPE = 0,
        CONVEX    = 1
} enumFaceType ;
```

Remarks

The Visual 3Space Control can import AutoCAD DXF files. The scene graph produced by the import process contains a Shape Hints node that provides information that helps Open Inventor improve rendering performance. The dxfFaceType property specifies the face type for the Shape Hints node. The default value of dxfFaceType is UNKNOWN_FACE_TYPE.

See Also

dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfCreaseAngle

dxfShapeType

DXF import shape type.

Possible Values

```
typedef enum {
        UNKNOWN_SHAPE_TYPE = 0,
        SOLID= 1
} enumShapeType ;
```

Remarks

The Visual 3Space Control can import AutoCAD DXF files. The scene graph produced by the import process contains a Shape Hints node that provides information that helps Open Inventor improve rendering performance. The dxfShapeType property specifies the shape type for the Shape Hints node. The default value of dxfShapeType is UNKNOWN_SHAPE_TYPE.

See Also

dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfFaceType
dxfCreaseAngle

dxfSpecularB

DXF import specular blue component.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffuse color only. The specular color for each imported entity is specified by the dxfSpecular[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfSpecularG

DXF import specular green component.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffuse color only. The specular color for each imported entity is specified by the dxfSpecular[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfSpecularR

DXF import specular red component.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The Visual 3Space Control can import AutoCAD DXF files. DXF entities specify a diffuse color only. The specular color for each imported entity is specified by the dxfSpecular[RGB] properties.

See Also

dxfAmbientScaleR
dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularG
dxfSpecularB
dxfVertexOrdering
dxfShapeType
dxfFaceType
dxfCreaseAngle

dxfVertexOrdering

DXF import vertex ordering.

Possible Values

```
typedef enum {
        UNKNOWN_ORDERING = 0,
        CLOCKWISE           = 1,
        COUNTERCLOCKWISE = 2
} enumVertexOrdering ;
```

Remarks

The Visual 3Space Control can import AutoCAD DXF files. The scene graph produced by the import process contains a Shape Hints node that provides information that helps Open Inventor improve rendering performance. The dxfVertexOrdering property specifies the vertex ordering for the Shape Hints node. The default value of dxfVertexOrdering is COUNTERCLOCKWISE.

See Also

dxfAmbientScaleG
dxfAmbientScaleB
dxfEmissiveScaleR
dxfEmissiveScaleG
dxfEmissiveScaleB
dxfSpecularR
dxfSpecularG
dxfSpecularB
dxfShapeType
dxfFaceType
dxfCreaseAngle

editCopy

Copies the current selection to the clipboard.

void editCopy()

Remarks

If nothing is selected, this method has no affect.

See Also

editCut
editDelete
editPaste
editSelectAll
getNumSelected
isSelectionEnable
selectionEnable
setSelectionMethod

editCut

Copies the current selection to the clipboard and then deletes the current selection.

void editCut()

Remarks

If nothing is selected, this method has no affect.

See Also

editCopy
editDelete
editPaste
editSelectAll
getNumSelected
isSelectionEnable
selectionEnable
setSelectionMethod

editDelete

Deletes the current selection.

void editDelete()

Remarks

If nothing is selected, this method has no affect.

See Also

editCopy
editCut
editPaste
editSelectAll
getNumSelected
isSelectionEnable
selectionEnable
setSelectionMethod

editPaste

Places the contents of the clipboard into the current scene graph.

void editPaste()

See Also

editCopy
editCut
editDelete
editSelectAll
getNumSelected
isSelectionEnable
selectionEnable
setSelectionMethod

editSelectAll

Selects the entire scene graph.

void editSelectAll()

See Also

editCopy
editCut
editDelete
editPaste
getNumSelected
isSelectionEnable
selectionEnable
setSelectionMethod

extendedPopupEnable

Enable/Disable extended popup menu.

Possible Values

TRUE or FALSE.

Remarks

Set extendedPopupEnable to TRUE or FALSE to enable or disable the extended popup menu. The extended popup menu contains additional functions not available on the default Open Inventor viewer popup menu.

feedbackSize

The point of rotation feedback size in pixels.

Remarks

The feedback size only applies to the examiner viewer. It is ignored by the walk, fly and plane viewers.

See Also

feedbackVisibility
getFeedbackSize
isFeedbackVisible
setFeedbackSize
setFeedbackVisibility

feedbackVisibility

The point of rotation feedback visibility.

Possible Values

TRUE	Feedback is visible.
FALSE	Feedback is not visible.

Remarks

The feedback visibility only applies to the examiner viewer. It is ignored by the walk, fly and plane viewers.

See Also

feedbackSize
getFeedbackSize
isFeedbackVisible
setFeedbackSize
setFeedbackVisibility

firstViewPoint
Sets the current camera to the first view point

void firstViewPoint()

Return Value

None

Parameters

None

Remarks

Sets the current camera to the first view point.

See Also
addViewPoint
lastViewPoint
nextViewPoint
priorViewPoint
resetViewPoints
traverseViewPoints

fog

Enable/Disable fog.

Possible Values

TRUE or FALSE.

Remarks

Set fog to TRUE to simulate the atmospheric effect of fog.

See Also

setFog

getBufferingType

Returns the buffering type of the current viewer.

long getBufferingType(**void**)

Return Value

The buffering type of the current viewer.

Possible values are:

0 Single Buffered
1 Double Buffered
2 Interactive

Remarks

In single buffered mode, the image is rendered directly to the window. This results in flickering between redraws but uses less memory. In double buffered mode, the image is rendered to a back buffer and copied to the viewing window. This reduces flicker. Interactive buffer mode uses double buffering when the user is doing interactive work and single buffering otherwise.

See Also

setBufferingType

getCameraAspectRatio

Get the aspect ratio of a camera.

float getCameraAspectRatio(**long** *nCameraId*)

Return Value

The aspect ratio of the camera specified by nCameraId.

Parameters

nCameraId ID of camera for which the aspect ratio is to be retrieved.

Remarks

The camera aspect ratio is the ratio of the viewing width to height. This value is greater than 0.0.

See Also

setCameraAspectRatio
createPerspectiveCamera

getCameraFarDistance

Get the far clipping plane distance of a camera.

float getCameraFarDistance(**long** *nCameraId*);

Return Value

The distance to the far clipping plane.

Parameters

nCameraId ID of camera for which the far distance is to be retrieved.

Remarks

The far clipping plane distance is measured from the viewpoint to the far clipping plane.

See Also

setCameraFarDistance
setCameraNearDistance
getCameraNearDistance
createPerspectiveCamera

getCameraFocalDistance

Get the focal distance of a camera.

float getCameraFocalDistance(**long** *nCameraId*);

Return Value

The distance to the far clipping plane.

Parameters

nCameraId ID of camera for which the focal distance is to be retrieved.

Remarks

The focal distance is the distance from the viewpoint to the point of focus. This value is ignored during rendering, but is used by some viewers to define a point of interest.

See Also

createPerspectiveCamera

getCameraHeight

Get the height angle of a perspective camera or the height of a orthographic camera.

float getCameraHeightAngle(**long** *nCameraId*);

Return Value

The camera height angle or the camera height.

Parameters

nCameraId ID of camera for which the height angle is to be retrieved.

Remarks

The value returned depends on the type of the camera specified by nCameraId. The height angle of a perspective camera determines the vertical angle of the perspective view volume.

See Also

createPerspectiveCamera
setCameraHeight
getCameraType

getCameraName

Get the name of a camera.

BSTR getCameraName(**long** *nCameraId*);

Return Value

A string containing the name of the camera.

Parameters

nCameraId ID of camera for which the name is to be retrieved.

Remarks

Each camera can have an associated string that is specified when the camera is created.

See Also

createPerspectiveCamera
setCameraName

getCameraNearDistance

Get the near clipping plane distance of a camera.

float getCameraNearDistance(**long** *nCameraId*);

Return Value

The distance to the near clipping plane.

Parameters

nCameraId ID of camera for which the near distance is to be retrieved.

Remarks

The near clipping plane distance is measured from the viewpoint to the near clipping plane.

See Also

createPerspectiveCamera
setCameraNearDistance
setCameraFarDistance
getCameraFarDistance

getCameraOrientation

Get the orientation of a camera.

void getCameraOrientation(**long** *nCameraId,*
float* *fAxisX*,
float* *fAxisY*,
float* *fAxisZ*,
float* *fAngleRad*)

Return Value

None.

Parameters

nCameraId ID of camera for which the orientation is to be retrieved
fAxisX address of float where the x component of the rotation axis is to be stored
fAxisY address of float where the y component of the rotation axis is to be stored
fAxisZ address of float where the z component of the rotation axis is to be stored
fAngleRad address of float where the rotation angle is to be stored

Remarks

The camera orientation is defined as the rotation of the viewing direction from the default direction (0,0,-1). To obtain the viewing direction of a camera, rotate the vector (0,0,-1) about the axis (fAxisX,fAxisY,fAxisZ) by fAngleRad radians.

See Also

createPerspectiveCamera
setCameraOrientation

getCameraPosition

Get the position of a camera.

void getCameraPosition(**long** *nCameraId,*
float* *fX*,
float* *fY*,
float* *fZ*)

Return Value

None.

Parameters

nCameraId ID of camera for which the orientation is to be retrieved.
fAxisX x coordinate of camera position
fAxisY y coordinate of camera position
fAxisZ z coordinate of camera position

Remarks

The camera position is defined as the location of the camera viewpoint.

See Also

createPerspectiveCamera
setCameraPosition

getCameraScaleHeight

Returns the scale height of a camera.

float getCameraScaleHeigt(**long** *nCameraId*

Return Value

The camera scale height.

Parameters

nCameraId	ID of camera to modify.

Remarks

Perspective cameras scale their height angle.

See Also

setCameraScaleHeight

getCameraType

Get the type of a camera.

long getCameraType(**long** *nCameraId*)

Return Value

The camera type. Possible values are:

0 Perspective Camera
1 Orthographic Camera

Parameters

nCameraId ID of camera for which the type is to be retrieved.

Remarks

Currently only perspective cameras are supported by the Visual 3Space Control.

See Also

createPerspectiveCamera

getCurrCameraId

Get the id of the currently active camera..

long getCurrCameraId(**void**)

Return Value

The ID of the currently active camera.

See Also

createPerspectiveCamera
setCurrentCamera

getFeedbackSize

Get the point of rotation feedback size in pixels.

long getFeedbackSize(**void**)

Return Value

The point of rotation feedback size in pixels.

Remarks

The feedback size only applies to the examiner viewer. It is ignored by the walk and fly viewers.

See Also

feedbackSize
feedbackVisibility
isFeedbackVisible
setFeedbackSize
setFeedbackVisibility

getFirstCamera

Returns the ID of the first camera.

long getFirstCamera(**void**)

Return Value

ID of first camera.

Remarks

The ID of the first camera is returned. If no cameras exist, getFirstCamera returns -1.

See Also

createPerspectiveCamera
getNextCamera

getFirstLight

Returns the ID of the first light.

long getFirstLight(**void**)

Return Value

ID of first light.

Remarks

The ID of the first light is returned. If no lights exist, getFirstLight returns -1.

See Also

getNextLight
createDirectLight
createPointLight
createSpotLight

getInteractiveDrawStyle

Get the still draw style of the viewer.

long getInteractiveDrawStyle(**void**)

Return Value

The interactive draw style. Possible values are:

0 As Is
1 Hidden Line (render only the front most lines)
2 No Texture (render without texture)
3 Low Complexity (render low complexity and no texture)
4 Line (wireframe)
5 Point (points only)
6 Bounding Box (draws the a wireframe bounding box)
7 Low Res Line (low complexity wireframe and no depth clearing)
8 Low Res Point (low complexity points and no depth clearing)
9 Same As Still (uses the current still draw style)

Remarks

The interactive draw style is specified independently of the still draw style. The interactive draw style determines the appearance of the scene while the scene is changing.

See Also

setInteractiveDrawStyle
setStillDrawStyle
getStillDrawStyle

getLightColor

Get the RGB color of a light.

void getLightColor(**long** *lLightId,*
float* *lpfRed*,
float* lp*fGreen*,
float* *lpfBlue*)

Return Value

None.

Parameters

lLightId	ID of camera for which the color is to be retrieved
lpfRed	address of float where the red component of the color is to be stored
lpfGreen	address of float where the green component of the color is to be stored
fpfBlue	address of float where the blue component of the color is to be stored

Remarks

Get the light source illumination color. Component colors range from 0.0 to 1.0

See Also

setLightColor

getLightDirection

Get the illumination direction vector of a light source.

void getLightDirection(**long** *lLightId,*
float* lp*fX,*
float* *lpfY,*
float* *lpfZ*)

Return Value

None.

Parameters

lLightId	ID of light for which the direction vector is to be retrieved
lpfX	Address of the float into which the X component of the vector is retrieved.
lpfY	Address of the float into which the Y component of the vector is retrieved.
lpfZ	Address of the float into which the Z component of the vector is retrieved.

Remarks

Only Direct and Spot lights have a direction.

See Also

setLightDirection

getLightDropOffRate

Get the drop off rate of a spot light.

void getLightDropOffRate(**long** *lLightId,*
float* *lpfDropRate*)

Return Value

None.

Parameters

lLightId	ID of light for which the drop off rate is retrieved.
lpfDropRate	Address of the float into which the rate is retrieved.

Remarks

Only Spot lights have a drop off rate. The rate measures the intensity drop off per change in angle from the primary direction: 0 = constant intensity, 1 = very sharp drop off.

See Also

setLightDropOffRate

getLightId

Returns the light id of the first light source with the same name..

long getLightId(**BSTR** *lpszLightName*)

Return Value

ID of first light with the same name. If no light is found getLightId returns -1.

Remarks

If more than one light exists with the same name, the first light id is returned.

See Also

setLightName

getLightIntensity

Get the illumination intensity of a light source.

void getLightIntensity(**long** *lLightId,*
float* lp*fIntensity*)

Return Value

None.

Parameters

lLightId	ID of light for which the intensity is to be retrieved
lpfIntensity	address of float where the intensity of the light is to be stored

Remarks

Vaild values range from 0.0 (no illumination) to 1.0 (maximum illumination)

See Also

setLightIntensity

getLightLocation

Get the location of a light source.

void getLightLocation(**long** *lLightId,*
float* lp*fX,*
float* *lpfY,*
float* *lpfZ*)

Return Value

None.

Parameters

lLightId	ID of light for which the location is to be retrieved
lpfX	Address of the float into which the X component is retrieved.
lpfY	Address of the float into which the Y component is retrieved.
lpfZ	Address of the float into which the Z component is retrieved.

Remarks

Only Point and Spot lights have a location.

See Also

setLightLocation

getLightName

Get the name of a light.

BSTR getLightName(**long** *lLightId*);

Return Value

A string containing the name of the light.

Parameters

lLightId ID of light for which the name is to be retrieved.

Remarks

Each light can have an associated string that is specified when the camera is created.

See Also

setLightName

getLightType

Get the type of the target light resource.

void getLightType(**long** *lLightId*)

Return Value

1 Light is a directional light.
2 Light is a point light.
3 Light is a spot light.
-1 Unknown light type or error.

Parameters

lLightId ID of light for which the type is to be retrieved

See Also

createDirectLight
createPointLight
createSpotLight

getNextCamera

Returns the ID of the next camera after the current camera.

long getNextCamera(**void**)

Return Value

ID of next camera.

Remarks

The ID of the next camera after the current camera is returned. If no cameras exist, getNextCamera returns -1.

See Also

createPerspectiveCamera
getFirstCamera

getNextLight

Returns the ID of the next light after the current light.

long getNextLight(**void**)

Return Value

ID of next light.

Remarks

The ID of the next light after the current light is returned. If no light exist, getNextLight returns -1.

See Also

getFirstLight
createDirectLight
createPointLight
createSpotLight

getNumCameras

Get the number of defined cameras.

long getNumCameras(**void**)

Return Value

The number of cameras that have been created.

See Also

createPerspectiveCamera

getNumSelected

Returns the number of paths in the selection list.

long getNumSelected()

Return Value

The number of paths in the selection list.

Remarks

If nothing is selected, this method returns 0.

See Also

editCopy
editCut
editDelete
editPaste
editSelectAll

getSeekTime

Get the seek time.

float getSeekTime(**void**)

Return Value

The seek time in seconds.

Remarks

The seek time is the time a seek takes to change to the new camera location.

See Also

setSeekTime
setDetailSeek
isDetailSeek

getStillDrawStyle

Get the still draw style of the viewer.

long getStillDrawStyle(**void**)

Return Value

The still draw style. Possible values are:

0 As Is
1 Hidden Line (render only the front most lines)
2 No Texture (render without texture)
3 Low Complexity (render low complexity and no texture)
4 Line (wireframe)
5 Point (points only)
6 Bounding Box (draws the a wireframe bounding box)
7 Low Res Line (low complexity wireframe and no depth clearing)
8 Low Res Point (low complexity points and no depth clearing)

Remarks

The still draw style is specified independently of the interactive draw style.

See Also

setStillDrawStyle
setInteractiveDrawStyle
getInteractiveDrawStyle

getViewerType

Get the current viewer type.

long getViewerType(**void**)

Return Value

Value indicating the current viewer type. Possible values are:

0 Examiner Viewer
1 Fly Viewer
2 Walk Viewer
3 Plane Viewer

See Also
setExaminerViewer
setFlyViewer
setPlaneViewer
setViewerType
setWalkViewer
SelectViewer
viewerType

headlightOn

Turn the headlight on or off for the scene.

Possible Values

TRUE	The headlight is on.
FALSE	The headlight is off.

Remarks

The headlight is the default light for the scene.

See Also

isHeadlight
setHeadlight

hideAllLights

Hide all the lights in the scene graph

VOID hideAllLights(**void**)

Return Value

None.

Remarks

Hides the light icons. While hidden, the lights still illuminate the scene.

See Also

showAllLights

importSceneGraph

The importSceneGraph method loads a scene from an Open Inventor file.

BOOL readSceneGraph(
    LPCTSTR lpszFileName
    );

Parameters

lpszFileName the name of the file.

Return Values

Returns TRUE if the scene was loaded, otherwise FALSE.

Remarks

A scene graph is loaded from a file. The scene graph is added to the scene without replacing the current scene graph.

See Also

ImportFile
OpenFile
SaveFile
deleteSceneGraph
readSceneGraph
readSceneGraphEx
importSceneGraphEx
serializeSceneGraphEx
saveSceneGraph
isValidFile
isValidFileEx

importSceneGraphEx

Import a scene from a file.

BOOL importSceneGraphEx(
LPCTSTR lpszFileName,
long nFileType
);

Parameters

lpszFileName the name of the file.
nFileType the format of the file containing the scene, possible values are:

```
typedef enum {
        VRML_OR_INVENTOR=0,
        AUTOCAD_DXF=1
} enumFileFormat;
```

Return Values

Returns TRUE if the scene was loaded, otherwise FALSE.

Remarks

A scene graph is imported from a file. The scene graph read from the file is appended to the current scene graph.

See Also

ImportFile
OpenFile
SaveFile
readSceneGraph
readSceneGraphEx
importSceneGraphEx
serializeSceneGraphEx
deleteSceneGraph
saveSceneGraph
isValidFile
isValidFileEx

interactiveDrawStyle

The interactive draw style of the viewer.

Possible Values

0 As Is
1 Hidden Line (render only the front most lines)
2 No Texture (render without texture)
3 Low Complexity (render low complexity and no texture)
4 Line (wireframe)
5 Point (points only)
6 Bounding Box (draws the a wireframe bounding box)
7 Low Res Line (low complexity wireframe and no depth clearing)
8 Low Res Point (low complexity points and no depth clearing)
9 Same As Still (uses the current still draw style)

Remarks

The interactive draw style is specified independently of the still draw style. The interactive draw style determines the appearance of the scene while the scene is changing.

See Also

setInteractiveDrawStyle
getInteractiveDrawStyle
setStillDrawStyle
getStillDrawStyle

isAutoClipping

Is autoclipping on of off.

BOOL isAutoClipping()

Return Value

TRUE	Autoclipping is on.
FALSE	Autoclipping is off.

Parameters

None.

Remarks

When on, the near and far camera clipping planes are continuously adjusted around the scene's bounding box to minimize clipping. Autoclipping is on by default.

See Also

autoClipping
setAutoClipping

isDecorating

Are the scene viewers decorations being displayed.

BOOL isDecorating(**void**)

Return Value

True	The fly wheels are visible.
False	The fly wheels are not visible.

isDetailSeek

Is viewer detail seek enabled.

BOOL isDetailSeek(**void**)

Return Value

True	Detail seek is enbled.
False	Detail seek is not enabled.

See Also

detailSeekOn
setDetailSeek

isFeedbackVisible

Is the point of rotation feedback visible.

BOOL isFeedbackVisible(**void**)

Return Value

TRUE	Feedback is visible.
FALSE	Feedback is not visible.

Remarks

The feedback size only applies to the examiner viewer. It is ignored by the walk, fly and plane viewers.

See Also

feedbackSize
feedbackVisibility
getFeedbackSize
setFeedbackSize
setFeedbackVisibility

isHeadlight

Is the headlight for the scene turned on ?.

BOOL isHeadlight(**void**)

Return Value

TRUE	The headlight is on.
FALSE	The headlight is off.

Remarks

The headlight is the default light for the scene.

See Also

headlightOn
setHeadlight

isLightOn

Is the light on or off.

BOOL isLightOn(**long** *lLightId)*

Return Value

TRUE	light is on.
FALSE	light is off.

Parameters

lLightId	ID of light for which the light on/off status is to be retrieved

Remarks

See Also

turnOffLight
turnOnLight

isPopupMenuEnabled

Is the viewer popup menu enabled.

BOOL isPopupMenuEnabled(**void**)

Return Value

True	The viewer popup menu is enabled.
False	The viewer popup menu is not enabled.

See Also

setPopupMenuEnabled

isSelectionEnable

Is selection enabled.

BOOL isSelectionEnable(**void**)

Return Value

True	Selection is enabled.
False	Selection is not enabled.

See Also

setSelectionEnable
selectionEnable
setSelectionMethod
setNoneManip
setCenterballManip
setJackManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

isUrlPickEnable

Is URL picking enabled.

BOOL isUrlPickEnable(**void**)

Return Value

True	URL picking is enabled.
False	URL picking is not enabled.

Remarks

Refer to urlPickEnable for a description of URL picking.

See Also

urlPickEnable
setUrlPickEnable
UpdatePickUrlName

isValidFile

Is the contents of the file understood by the control.

BOOL isValidFile(**LPCTSTR** lpszFileName)

Return Value

TRUE	File is understood.
FALSE	File is not understood.

Parameters

lpszFileName	the name of the file.

Remarks

Call this method to determine if the control can load a file. The Visual 3Space Control currently supports Open Inventor, VRML 1.0, and AutoCAD DXF file formats.

See Also

ImportFile
OpenFile
SaveFile
readSceneGraph
readSceneGraphEx
importSceneGraph
importSceneGraphEx
serializeSceneGraphEx
isValidFileEx

isValidFileEx

Is the contents of the archive understood by the control.

BOOL isValidFile(**VARIANT *** pArchive)

Return Value

TRUE	Archive contents are understood.
FALSE	Archive contents are not understood.

Parameters

pArchive	a pointer to an MFC CArchive object.

Remarks

Call this method to determine if the control can load the contents of an MFC CArchive object. The Visual 3Space Control currently supports Open Inventor, VRML 1.0, and AutoCAD DXF file formats.

See Also

ImportFile
OpenFile
SaveFile
readSceneGraph
readSceneGraphEx
importSceneGraph
importSceneGraphEx
serializeSceneGraphEx
isValidFile

isViewing

Is viewing mode on or off.

BOOL isViewing()

Return Value

TRUE	The viewer is on.
FALSE	The viewer is off..

Parameters

None.

Remarks

Refer to viewingOn for a description of viewing mode.

See Also

setViewing
viewingOn

isWWWEnable

Is the HTML browser DDE remote control interface enabled.

BOOL isWWWEnable(**void**)

Return Value

True	Remote control is enabled.
False	Remote conrol is not enabled.

Remarks

Refer to wwwEnable for a description of browser remote control.

See Also

setWWWEnable
wwwEnable
urlPickEnable

lastFileName

Name of the last file loaded.

Possible Values

A valid BSTR that conatains the path of the last file loaded.

Remarks

If the remberLastFile property is TRUE, the Visual 3Space Control updates this property each time it loads a file. If rememberLastFile is TRUE and serializeSceneGraph is FALSE, this property supercedes startFileName . You should not modify this property.

See Also

rememberLastFile
startFileName
serializeSceneGraph
lastFileType

lastFileType

The file format of the last file loaded.

Possible Values

```
typedef enum {
        VRML_OR_INVENTOR=0,
        AUTOCAD_DXF=1
} enumFileFormat;
```

Remarks

This property contains a value that indicates the type of the last file that was read by the Visual 3Space Control. You should not modify this property.

See Also

rememberLastFile
startFileName
serializeSceneGraph
lastFileName

lastViewPoint

Sets the current camera to the last view point

void lastViewPoint()

Return Value

None

Parameters

None

Remarks

Sets the current camera to the last view point.

See Also

addViewPoint
firstViewPoint
nextViewPoint
priorViewPoint
resetViewPoints
traverseViewPoints

manipulatorType

Viewer Manipulator type.

Possible Values

```
typedef enum {
        None=0,
        TrackBall=1
        HandleBox=2,
        Jack=3,
        CenterBall=4,
        TransformBox=5,
        TabBox=6
} enumManipType;
```

Remarks

Sets the manipulator type used to manipulate selected objects.

See Also

setManipNone
setCenterballManip
setHandleboxManip
setJackManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

nextViewPoint

Sets the current camera to the next view point after the current view point.

void nextViewPoint()

Return Value

None

Parameters

None

Remarks

Position to the next view point.

See Also

addViewPoint
firstViewPoint
lastViewPoint
nextViewPoint
priorViewPoint
resetViewPoints
traverseViewPoints

pointCameraAt

Sets the orientation of the camera so that it points toward the given target point while keeping the "up" direction of the camera parallel to the positive y-axis. If this is not possible, it uses the positive z-axis as "up".

void pointCameraAt(long nCameraId,
float fX,
float fY,
float fZ)

Return Value

None.

Parameters

nCameraId	ID of the camera which is being pointed at the target location.
fX	Target X coordinate.
fY	Target Y coordinate.
fZ	Target Z coordinate.

See Also

setCameraPosition
getCameraPosition
setCameraOrientation
getCameraOrientation

popupMenuEnabled

priorViewPoint

Sets the current camera to the view point before the current view point.

void priorViewPoint()

Return Value

None

Parameters

None

Remarks

Position to the next view point.

See Also

addViewPoint
firstViewPoint
lastViewPoint
nextViewPoint
resetViewPoints
traverseViewPoints

readSceneGraph

The readSceneGraph method loads a scene from an Open Inventor file.

BOOL readSceneGraph(
    LPCTSTR lpszFileName
    );

Parameters

lpszFileName the name of the file.

Return Values

Returns TRUE if the scene was loaded, otherwise FALSE.

Remarks

A scene graph is loaded from a file. The current scene graph is replaced by the scene graph loaded from the file.

See Also

ImportFile
OpenFile
SaveFile
readSceneGraphEx
importSceneGraph
importSceneGraphEx
serializeSceneGraphEx
deleteSceneGraph
saveSceneGraph
isValidFile
isValidFileEx

readSceneGraphEx

Read a scene from a file.

BOOL readSceneGraphEx(
LPCTSTR lpszFileName,
long nFileType
);

Parameters

lpszFileName the name of the file.
nFileType the format of the file containing the scene, possible values are:

```
typedef enum {
        VRML_OR_INVENTOR=0,
        AUTOCAD_DXF=1
} enumFileFormat;
```

Return Values

Returns TRUE if the scene was loaded, otherwise FALSE.

Remarks

A scene graph is loaded from a file. The current scene graph is replaced by the scene graph loaded from the file.

See Also

ImportFile
OpenFile
SaveFile
readSceneGraph
importSceneGraph
importSceneGraphEx
serializeSceneGraphEx
deleteSceneGraph
saveSceneGraph
isValidFile
isValidFileEx

rememberLastFile

Enable memory of last file loaded.

Possible Values

True	Memory enabled.
False	Memory disabled.

Remarks

When remberLastFile is TRUE, the Visual 3Space control will remember each file that is loaded. If serializeSceneGraph is FALSE, each time a file is loaded the lastFileName property will be updated. When the container application exits and then restarts, the file name contained in lastFileName will be used in preference to startFileName. If serializeSceneGraph is TRUE, then the scene graph is serialized along with the control's other persistent data (i.e. properties). Using combinations of rememberLastFile and serializeSceneGraph, you can control whether the Visual 3Space Control stores a link to an external data source (scene graph file), or embeds the scene graph inside storage provided by the containing application.

See Also

lastFileName
startFileName
serializeSceneGraph

resetToHomePosition

Set the current camera values to the last home position.

void resetToHomePosition()

Return Value

None.

Parameters

None.

Remarks

Set the camera position to the previously saved home position.

See Also

saveHomePosition

resetViewPoints

Clears the view point list.

void resetViewPoints()

Return Value

None

Parameters

None

Remarks

All view points entered using addViewPoint are discarded.

See Also

addViewPoint
firstViewPoint
lastViewPoint
nextViewPoint
priorViewPoint
traverseViewPoints

saveHomePosition

Saves the current camera values.

void saveHomePosition()

Return Value

None.

Parameters

None.

Remarks

Saves the current camera values so that the camera can quickly be reset to this position later.

See Also

resetToHomePosition

saveSceneGraph

The saveSceneGraph method saves the scene to a an Open Inventor file.

long saveSceneGraph(
    LPCTSTR lpszFileName
    );

Parameters

lpszFileName the name of the file.

Return Values

none.

Remarks

The scene graph is saved in a file.

See Also

ImportFile
importSceneGraph
OpenFile
SaveFile
deleteSceneGraph
importSceneGraph
readSceneGraph

seekTime

selectionEnable

Enables or disables selection for the viewer.

Possible Values

TRUE, turn selection on.
FALSE, turn selection off.

Remarks

When selection is enabled, components of the scene graph can be selected by picking with the mouse pointer. The selection method determines how scene graph components are selected. When a component of the scene graph has been selected, it is highlighted by drawing a wireframe bounding box around it if the current manipulator is set to none, otherwise the current manipulator is drawn around the selected component.

See Also

setSelectionEnable
isSelectionEnable
setSelectionMethod
setNoneManip
setCenterballManip
setJackManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip
viewingOn

selectionMethod

Sets the selection method for the viewer.

Possible Values

0 - SINGLE selection policy
1 - TOGGLE selection policy
2 - SHIFT selection policy

Remarks

The selection method determines how selection is performed.

SINGLE
Left mouse pick on object clears selection, then selects object. Left mouse pick on nothing clears selection. Only one object may be selected at a time.

TOGGLE
Left mouse pick on object toggles its selection status. Left mouse pick on nothing does nothing. Multiple objects may be selected.

SHIFT
When shift key is down, selection policy is TOGGLE. When shift key is up, selection policy is SINGLE. Multiple objects may be seleted.

The default method is SHIFT.

See Also

isSelectionEnable
selectionEnable
setSelectionMethod

serializeControl

Serialize the control to an MFC Archive Object.

void serializeControl(Variant * *pArchive*)

Return Value

None.

Parameters

pArchive - a pointer to an MFC CArchive object.

Remarks

Use this method to serialize the Visual 3Space Control to or from an MFC CArchive object.

serializeSceneGraph

Enable serialization of the scene graph.

Possible Values

True	Serialization enabled.
False	Serialization disabled.

Remarks

When serializeSceneGraph and rememberLastFile are both TRUE, the Visual 3Space Control will remember each file that is loaded. The scene graph will be stored as a BLOB (Binary Large OBject) along with the rest of the control's persistent state (i.e. properties). For example, consider a control that has been embedded in a Microsoft Word Document. If you set serializeSceneGraph and rememberLastFile to TRUE, load a scene graph file (Open Inventor or VRML) into the control, and save the document, the scene graph will be stored inside the Microsoft Word Document along with all of the control's properties. If you move the Microsoft Word Document to a different system that does not contain the original scene graph file, when you open the document in Microsoft Word the scene graph is restored. In contrast, setting rememberLastFile TRUE and serializeSceneGraph FALSE will instruct the Visual 3Space Control to store a link to the scene graph file. If you moved the Word Document to a different system that did not contain the scene graph file in the same location as the original system, then when you opened the Word Document, the scene graph would not be restored. By using combinations of rememberLastFile and serializeSceneGraph, the Visual 3Space Control will embed or link the scene graph data:

rememberLastFile	serializeSceneGraph	
TRUE	TRUE	scene graph is embedded
TRUE	FALSE	link to scene graph stored
FALSE	TRUE	no link, no embedding
FALSE	FALSE	no link, no embedding

See Also

lastFileName
startFileName
rememberLastFile

serializeSceneGraphEx

Serialize scene graph to/from an MFC Archive Object.

BOOL serializeSceneGraphEx(
VARIANT * pArchive,
boolean bVrmlHeader
);

Parameters

lpszFileName	the name of the file.
bVrmlHeader	flag indicating header to use when storing.

Return Values

Returns TRUE if the scene was loaded/stored successfully, otherwise FALSE.

Remarks

Use this method to serialize the scene graph to/from an MFC CArchive object.

See Also

ImportFile
OpenFile
SaveFile
readSceneGraph
readSceneGraphEx
importSceneGraph
importSceneGraphEx
deleteSceneGraph
saveSceneGraph
isValidFile
isValidFileEx

setAntialiasing

Turn antialiasing on or off.

void setAntialiasing(BOOL *nOnOff*)

Return Value

None.

Parameters

nOnOff TRUE, turn on antialiasing. FALSE, turn off antialiasing.

Remarks

Antialiasing techniques reduce jagged lines and make the objects drawn on the screen appear smooth.

See Also

antiAliasing
setStillDrawStyle
setInteractiveDrawStyle
getStillDrawStyle
getInteractiveDrawStyle

setAutoClipping

Turns autoclipping on of off.

void setAutoClipping(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn on autoclipping. FALSE, turn off autoclipping.

Remarks

When on, the near and far camera clipping planes are continuously adjusted around the scene's bounding box to minimize clipping. Autoclipping is on by default.

See Also

autoClipping
isAutoClipping

setBufferingType

Sets the buffering type of the current viewer.

void setBufferingType(**long** *nType*)

Return Value

None.

Parameters

nType The buffering type.

Possible values are:

0 Single Buffered
1 Double Buffered
2 Interactive

Remarks

In single buffered mode, the image is rendered directly to the window. This results in flickering between redraws but uses less memory. In double buffered mode, the image is rendered to a back buffer and copied to the viewing window. This reduces flicker. Interactive buffer mode uses double buffering when the user is doing interactive work and single buffering otherwise.

See Also

getBufferingType

setCameraAspectRatio

Set the aspect ratio of a camera.

void setCameraAspectRatio(**long** *nCameraId*, **float** *fAspectRatio*)

Return Value

None.

Parameters

nCameraId ID of camera to modify
fAspectRatio The camera aspect ratio to set

Remarks

The camera aspect ratio is the ratio of the viewing width to height. This value must be greater than 0.0.

See Also

getCameraAspectRatio
createPerspectiveCamera

setCameraFarDistance

Set the far clipping plane distance of a camera.

void setCameraFarDistance(**long** *nCameraId*, **float** *fDist*)

Return Value

None.

Parameters

nCameraId ID of camera to modify.
fDist distance to far clipping plane.

Remarks

The far clipping plane distance is measured from the viewpoint to the far clipping plane.

See Also

getCameraFarDistance
getCameraNearDistance
setCameraNearDistance
createPerspectiveCamera

setCameraHeight

Set the height angle of a perspective camera or the height of a orthographic camera.

void setCameraHeight(**long** *nCameraId*, **float** *fHeight*)

Return Value

None.

Parameters

nCameraId ID of camera for which the height angle is to be retrieved.
fHeight The camera height angle or the camera height.

Remarks

The value specified by fHeight depends on the type of the camera specified by nCameraId. The height angle of a perspective camera determines the vertical angle of the perspective view volume.

See Also

createPerspectiveCamera
getCameraHeight
getCameraType

setCameraName

Set the name of a camera.

void setCameraName(**long** *nCameraId*, **BSTR** *lpszCameraName*)

Return Value

None.

Parameters

nCameraId ID of camera for which the name is to be set.
lpszCameraName A string containing the name of the camera.

Remarks

Each camera can have an associated string that is specified when the camera is created or by calling setCameraName.

See Also

createPerspectiveCamera
setCameraName

setCameraNearDistance

Set the near clipping plane distance of a camera.

void setCameraNearDistance(**long** *nCameraId*, **float** *fDist*)

Return Value

None.

Parameters

nCameraId ID of camera to modify.
fDist distance to near clipping plane.

Remarks

The far clipping plane distance is measured from the viewpoint to the near clipping plane.

See Also

getCameraNearDistance
getCameraFarDistance
setCameraFarDistance
createPerspectiveCamera

setCameraOrientation

Set the orientation of a camera.

void setCameraOrientation(**long** *nCameraId,*
float *fAxisX*,
float *fAxisY*,
float *fAxisZ*,
float *fAngleRad*);

Return Value

None.

Parameters

nCameraId ID of camera to modify
fAxisX x component of rotation axis
fAxisY y component of rotation axis
fAxisZ z component of rotation axis
fAngleRad rotation angle in radians

Remarks

The camera orientation is defined as the rotation of the viewing direction from the default direction (0,0,-1). To obtain the viewing direction of a camera, rotate the vector (0,0,-1) about the axis (fAxisX,fAxisY,fAxisZ) by fAngleRad radians.

See Also

getCameraOrientation
createPerspectiveCamera

setCameraPosition

Set the position of a camera.

void setCameraPosition(**long** *nCameraId,*
float *fX*,
float *fY*,
float *fZ*)

Return Value

None.

Parameters

nCameraId ID of camera of camera to modify
fAxisX x coordinate of camera position
fAxisY y coordinate of camera position
fAxisZ z coordinate of camera position

Remarks

The camera position is defined as the location of the camera viewpoint.

See Also

createPerspectiveCamera
getCameraPosition

setCameraScaleHeight

Set the scale height of a camera.

void setCameraScaleHeigt(**long** *nCameraId*, **float** *fHeight*)

Return Value

None.

Parameters

nCameraId ID of camera to modify.
fHeight The camera scale height.

Remarks

Scales the height of the camera. Perspective cameras scale their height angle.

See Also

getCameraScaleHeight

setCenterballManip

Surrounds the current selection with a centerball manipulator.

void setCenterballManip(void)

Return Value

None.

Parameters

None.

Remarks

The manipulator is shaped like a sphere defined by three intersecting circles. Where the circles intersect (at the ends of the x, y and z axes) there are sets of small green crosshairs. Dragging a pair of crosshairs translates the entire centerball within the plane of the crosshairs. Dragging a circle rotates about a constrained axis and dragging the area areas between them rotates the sphere freely about the center. An invisible but pickable sphere initiates the free-rotation dragging.

See Also

setJackManip
setNoneManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

setCurrentCamera

Sets the active camera for the scene viewer control to the target camera ID.

void setCurrentCamera(**long** *lCameraId)*

Return Value

None.

Parameters

lCameraId Target camera ID

See Also

createPerspectiveCamera
getCurrCameraId

setDetailSeek

Turns detail seek on of off.

void setDetailSeek(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn detail seek on. FALSE, turn detail seek off.

Remarks

Refert to detailSeekOn for a description of URL picking.

See Also

isDetailSeek
detailSeekOn

setExaminerViewer

Switch to the examiner viewer.

void setExaminerViewer(**void**)

Return Value

None.

See Also
getViewerType
setFlyViewer
setWalkViewer
setPlaneViewer
SelectViewer

setFeedbackSize

Set the point of rotation feedback size in pixels.

void setFeedbackSize(**long** *nFeedbackSize*)

Return Value

None.

Parameters

nFeedbackSize The point of rotation feedback size in pixels.

Remarks

The feedback size only applies to the examiner viewer. It is ignored by the walk, fly and plane viewers.

See Also

feedbackSize
feedbackVisibility
getFeedbackSize
isFeedbackVisible
setFeedbackVisibility

setFeedbackVisibility

Set the point of rotation feedback visibilty.

void setFeedbackVisibility(BOOL *nVisible*)

Return Value

None.

Parameters

nVisible TRUE or FALSE, the feedback is visible.

Remarks

Feedback visibility only applies to the examiner viewer. It is ignored by the walk, fly and plane viewers.

See Also

feedbackSize
feedbackVisibility
getFeedbackSize
isFeedbackVisible
setFeedbackSize

setFlyViewer

Switch to the fly viewer.

void setFlyViewer(**void**)

Return Value

None.

See Also

getViewerType
setExaminerViewer
setPlaneViewer
setWalkViewer
SelectViewer

setFog

Turns fog on or off for the scene.

void setFog(**BOOL** *nOnOff)*

Return Value

None.

Parameters

nOnOff TRUE, turn fog on. FALSE, turn fog off.

Remarks

Simulates the atmospheric effect of fog.

See Also

fog

setHandleboxManip

Surrounds the current selection with a handlebox manipulator.

void setHandleboxManip(void)

Return Value

None.

Parameters

None.

Remarks

The manipulator is shaped like a wireframe box with small corner cubes mounted on each corner. Click and drag any of these cubes to uniformly scale the current selection. While you drag a face of the box, purple feedback arrows display the possible directions of motion. Press the <Shift> key to constrain the motion to one of the two major directions in the plane. The constraint direction is chosen based on the next user gesture. Press the <ALT> key and the manipulator will translate perpendicular to that plane.

By default, dragging any of the small cubes scales about the center of the object. Pressing the <Alt> key changes this: A corner cube will scale about its opposite corner. A center cube will scale about the center of its opposite face.

See Also

setHeadlight

Turns the headlight on of off.

void setHeadlight(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn on headlight. FALSE, turn off headlight.

Remarks

The headlight is on by default.

See Also

isHeadlight

setInteractiveDrawStyle

Set the still draw style of the viewer.

void getInteractiveDrawStyle(**long** *nDrawStyle*)

Return Value

None.

Parameters

nDrawStyle The interactive draw style.

Possible values are:

0 As Is
1 Hidden Line (render only the front most lines)
2 No Texture (render without texture)
3 Low Complexity (render low complexity and no texture)
4 Line (wireframe)
5 Point (points only)
6 Bounding Box (draws the a wireframe bounding box)
7 Low Res Line (low complexity wireframe and no depth clearing)
8 Low Res Point (low complexity points and no depth clearing)
9 Same As Still (uses the current still draw style)

Remarks

The interactive draw style is specified independently of the still draw style. The interactive draw style determines the appearance of the scene while the scene is changing.

See Also

getInteractiveDrawStyle
setStillDrawStyle
getStillDrawStyle

setJackManip

Surrounds the current selection with a jack manipulator.

void setJackManip(void)

Return Value

None.

Parameters

None.

Remarks

The manipulator is in the shape of a jack from the children's game jacks. Three lines along the x, y, and z form the central star shape, which you can drag with the mouse to rotate the jack and selection. Dragging any of the small cubes mounted at the end of the axes will uniformly scale the manipulator and selection in all 3 dimensions.

See Also

setCenterballManip
setNoneManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip
manipulatorType

setLightColor

Set the RGB color of a light.

void setLightColor(**long** *lLightId,*
float *fRed*,
float *fGreen*,
float *fBlue*)

Return Value

None.

Parameters

lLightId	ID of camera for which the color is to be set.
fRed	the red component of the color.
fGreen	the green component of the color.
fBlue	the blue component of the color.

Remarks

Set the light source illumination color. Component colors range from 0.0 to 1.0

See Also

getLightColor

setLightDirection

Set the illumination direction vector of a light source.

void setLightDirection(**long** *lLightId,*
float f*X,*
float *fY,*
float *fZ*)

Return Value

None.

Parameters

lLightId	ID of light for which the direction vector is to be retrieved
fX	Float value for the X compnent of the vector.
fY	Float value for the Y component of the vector.
fZ	Float value for the Z component of the vector.

Remarks

Only Direct and Spot lights have a direction.

See Also

getLightDirection

setLightDropOffRate

Set the drop off rate of a spot light.

void setLightDropOffRate(**long** *lLightId,*
float *fDropRate*)

Return Value

None.

Parameters

lLightId	ID of light for which the drop off rate is set.
fDropRate	Float value of the drop off rate.

Remarks

Only Spot lights have a drop off rate. The rate measures the intensity drop off per change in angle from the primary direction: 0 = constant intensity, 1 = very sharp drop off.

See Also

getLightDropOffRate

setLightIntensity

Set the illumination intensity of a light.

void setLightIntensity(**long** *lLightId,*
float *fIntensity)*

Return Value

None.

Parameters

lLightId ID of light for which the intensity is to be set.
fIntensity *Illumination intensity.*

Remarks

Valid values range from 0.0 (no illumination) to 1.0 (maximum illumination).

See Also

getLightIntensity

setLightLocation

Set the location of a light source.

void getLightLocation(**long** *lLightId,*
float *fX,*
float *fY,*
float *fZ*)

Return Value

None.

Parameters

lLightId	ID of light for which the location is to be set.
fX	The X component of the location.
fY	The Y component of the location.
fZ	The Z component of the location.

Remarks

Only Point and Spot lights have a location.

See Also

getLightLocation

setLightName

Set the name of a light.

void setLightName(**long** *lLightId*, **BSTR** *lpszLightName*)

Return Value

None.

Parameters

lLigthtId ID of light for which the name is to be set.
lpszLightName A string containing the name of the light.

Remarks

Each light can have an associated string that is specified when the light is created or by calling setLightName.

See Also

getLightName

setManipNone

setNoneManip

Surrounds the current selection with a bounding box.

void setNoneManip(void)

Return Value

None.

Parameters

None.

Remarks

The bounding box serves to highlight the selection.

See Also

setCenterballManip
setJackManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

setPlaneViewer

Switch to the plane viewer.

void setPlaneViewer(**void**)

Return Value

None.

See Also
getViewerType
setExaminerViewer
setFlyViewer
setWalkViewer
SelectViewer

setPopupMenuEnabled

Turns the viewer popup menu on of off.

void setPopupMenuEnabled(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn popup menu on. FALSE, turn popup menu off.

See Also

isPopupMenuEnabled

setReplaceAllManip

Sets flag that indicates how to update the scene graph when the current manipulator is changed.

void setReplaceAllManip(void)

Return Value

None.

Parameters

None.

Remarks

If this flag is set to TRUE, then manipulators are replaced each time the type of manipulator is changed.

See Also

setCenterballManip
setNoneManip
setTabboxManip
setTrackballManip
setTransformBoxManip

setSeekTime

Set the seek time.

void setSeekTime(**float** *fSeekTime*)

Return Value

None.

Parameters

fSeekTime The seek time in seconds.

Remarks

The seek time is the time a seek takes to change to the new camera location.

See Also

getSeekTime
setDetailSeek
isDetailSeek

setSelectionEnable

Enables or disables selection for the viewer.

void setSelectionEnable(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn selection on. FALSE, turn selection off.

See Also

editCut
editDelete
editPaste
editSelectAll
getNumSelected
isSelectionEnable
selectionEnable
setSelectionMethod
setNoneManip
setCenterballManip
setJackManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

setSelectionMethod

Sets the selection method for the viewer.

void setSelectionMethod(***long*** *nSelectMethod*)

Return Value

None.

Parameters

nSelectMethod 0 - SINGLE selection policy
1 - TOGGLE selection policy
2 - SHIFT selection policy

Remarks

See selectionMethod for a description of selection methods.

See Also

isSelectionEnable
selectionEnable
selectionMethod

setStillDrawStyle

Set the still draw style of the viewer.

void setStillDrawStyle(**long** *nDrawStyle*)

Return Value

None.

Parameters

nDrawStyle The draw style to set

Possible values are:

0 As Is
1 Hidden Line (render only the front most lines)
2 No Texture (render without texture)
3 Low Complexity (render low complexity and no texture)
4 Line (wireframe)
5 Point (points only)
6 Bounding Box (draws the a wireframe bounding box)
7 Low Res Line (low complexity wireframe and no depth clearing)
8 Low Res Point (low complexity points and no depth clearing)

Remarks

The still draw style is specified independently of the interactive draw style.

See Also

getStillDrawStyle
setInteractiveDrawStyle
getInteractiveDrawStyle

setTabboxManip

Surrounds the current selection with a tabbox manipulator.

void setTabboxManip(void)

Return Value

None.

Parameters

None.

Remarks

The manipulator is shaped like a cube. 1D and 2D scaling are available for each face of the cube.

See Also

setCenterballManip
setJackManip
setNoneManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

setTrackballManip

Surrounds the current selection with a trackball manipulator.

void setTrackballManip(void)

Return Value

None.

Parameters

None.

Remarks

The manipulator is shaped like a ball and wraps around the selection in three circular stripes. The stripes are oriented like wheels that spin around the x, y, and z axes. Drag the stripes to rotate the selection around those axes. You do not have to hit the lines; pick anywhere within the stripe's outline. To rotate the trackball freely in 3 dimensions, click the area between the stripes and then drag. If the mouse is still moving when you release it, the trackball will continue to spin.

Press the <Alt> key to unifomly scale the trackball.

Press the <Shift> key and the user axis appears; this is a draggable axis with an extra stripe around it. Moving the mouse along the surface of the drags the "pole" of the axis. Release the <Shift> key and the user axis remains; drag the new stripe for constrained rotation around the user axis. To make the user axis disappear, press <Shift> and drag the pole to where two of the other stripes intersect. This aligns the user axis with the primary axis, at which point the user axis disappears.

See Also

setCenterballManip
setJackManip
setNoneManip
setTabboxManip
setTransformBoxManip
setReplaceAllManip

setTrackballManip.

setTransformBoxManip

Surrounds the current selection with a transform box manipulator.

void setTransformBoxManip(void)

Return Value

None.

Parameters

None.

Remarks

The manipulator is shaped like a box with small cubes at the corners. Click and drag any of these cubes to uniformly scale the selection. Drag any edge of the box to rotate the selection about its center, along an axis parallel to that edge. Pick any face of the box for 2D translation in the plane of that face.

See Also

setCenterballManip
setJackManip
setNoneManip
setTabboxManip
setTrackballManip
setReplaceAllManip

setTransparencyType

Sets the transparency quality level to use when rendering.

void setTransparencyType(***long*** *nTranType*)

Return Value

None.

Parameters

nTranType 0 - SCREEN_DOOR
1 - ADD
2 - DELAYED_ADD
3 - SORTED_OBJECT_ADD
4 - BLEND
5 - DELAYED_BLEND
6 - SORTED_OBJECT_BLEND

Remarks

See the transparencyType property for a description of transparency types.

See Also

transparencyType

setUrlPickEnable

Turns URL picking on of off.

void setUrlPickEnable(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn URL picking on. FALSE, turn URL picking off.

Remarks

Refer to urlPickEnable for a description of URL picking.

See Also

isUrlPickEnable
urlPickEnable

setViewerType

Set the current viewer type.

void setViewerType(**long** *lViewerType*)

Return Value

None.

Parameters

Possible values are:

0 Examiner Viewer
1 Fly Viewer
2 Walk Viewer
3 Plane Viewer

See Also

getViewerType
setExaminerViewer
setFlyViewer
setPlaneViewer
setWalkViewer
SelectViewer
viewerType

setViewing

Turns the viewing mode on or off.

void setViewing(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn on viewing. FALSE, turn off viewing.

Remarks

Refer to viewingOn for a description of viewing mode.

See Also

isViewing
viewingOn

setWWWEnable

Enables/Disables the HTML browser remote control DDE interface.

void setWWWEnable(***BOOL*** *nOnOrOff*)

Return Value

None.

Parameters

nOnOrOff TRUE, turn remote control on. FALSE, turn remote control off.

Remarks

Refer to wwwEnable for a description of HTML browser remote control.

See Also

isWWWEnable
wwwEnable
urlPickEnable

setWalkViewer

Switch to the walk viewer.

void setWalkViewer(**void**)

Return Value

None.

See Also
getViewerType
setExaminerViewer
setFlyViewer
setPlaneViewer
SelectViewer

showAllLights

Show all the lights in the scene graph

VOID showAllLights(**void**)

Return Value

None.

Remarks

Displays all the light icons.

See Also

hideAllLights

startFileName

Name of file to load on startup.

Possible Values

A valid BSTR that contains a valid path name.

Remarks

If you want the Visual 3Space Control to load a scene graph file when the container application starts, set this property to the path of the scene graph file. This stores a link to the scene graph file. You must set serializeSceneGraph to FALSE in order for startFileName to be used, otherwise it is ignored.

See Also

lastFileName
rememberLastFile
serializeSceneGraph

stillDrawStyle

The still draw style of the viewer.

Possible values are:

0 As Is
1 Hidden Line (render only the front most lines)
2 No Texture (render without texture)
3 Low Complexity (render low complexity and no texture)
4 Line (wireframe)
5 Point (points only)
6 Bounding Box (draws the a wireframe bounding box)
7 Low Res Line (low complexity wireframe and no depth clearing)
8 Low Res Point (low complexity points and no depth clearing)

Remarks

The still draw style is specified independently of the interactive draw style.

See Also

setStillDrawStyle
getStillDrawStyle
setInteractiveDrawStyle
getInteractiveDrawStyle

stopAnimating

Stop animation if it is occuring.

void stop Animating()

Return Value

None.

Remarks

This method only affects the examiner viewer. It is ignored if the current viewer is not the examiner viewer.

See Also

setExaminerViewer

text3d

The 3D text string.

Possible Values

A BSTR that conatains a string to be used for 3D text.

Remarks

The Visual 3Space Control can be used to display 3D text. To display 3D text, set the text3d property to the 3D text you want to display, and then set the text3dEnable property to TRUE.

See Also

text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dBlue

3D text color (red component).

Possible Values
Floating point value between 0.0 and 1.0.

Remarks

The color of the 3D text is controled using the text3dRed, text3dGreen, and text3dBlue properties.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dComplexity

Complexity of 3D text.

Possible Values

Floating point value between 0.0 and 1.0.

Remarks

The text3dComplexity value controls the degree of tesselation of the polygon mesh produced to represent the 3D text string. A value of 0.0 produces minimum tesselation and a value of 1.0 produces maximum tesselation.

See Also

text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dFontName
text3dFontSize

text3dEnable

Enable/Disable 3D text.

Possible Values

TRUE or FALSE.

Remarks

The Visual 3Space Control can be used to display 3D text. To display 3D text, set the text3d property to the 3D text you want to display, and then set the text3dEnable property to TRUE. Setting text3dEnable to TRUE will destroy the current scene graph and replace it with the scene graph necessary to display the 3D text.

See Also

text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dFontName

Font name for 3D text.

Possible Values

String containing a TrueType font name and style name separated by a semi-colon. For example: "Times New Roman;Italic".

Remarks

The text3dFontName property controls the typeface and style of the font used to produce the 3D text. You must specify a TrueType font typeface name.

See Also

text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontSize

text3dFontSize

Font size for 3D text.

Possible Values

Floating point value.

Remarks

The text3dFontSize property specifies the size of the 3D text.

See Also

text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName

text3dGreen

3D text color (red component).

Possible Values
Floating point value between 0.0 and 1.0.

Remarks

The color of the 3D text is controled using the text3dRed, text3dGreen, and text3dBlue properties.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dJustification

Justification of 3D text.

Possible Values

```
typedef enum
{
	Left = 1,
	Right = 2,
	Center = 3
} enumText3dJustification;
```

Remarks

The text3dJustification property controls the justification of the 3D text.

See Also

text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dComplexity
text3dFontName
text3dFontSize

text3dParts

Parts of 3D text to display.

Possible Values

```
typedef enum
{
	Front = 1,
	Sides = 2,
	Back = 4,
	All = 7
} enumText3dParts;
```

Remarks

The text3dParts property controls which components of the 3D text are displayed.

See Also

text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dRed

3D text color (red component).

Possible Values
Floating point value between 0.0 and 1.0.

Remarks

The color of the 3D text is controled using the text3dRed, text3dGreen, and text3dBlue properties.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dRotationX

3D text x axis rotation angle (radians).

Possible Values
Floating point value.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. This value specifies the increment by which the text is rotated about the x axis.

See Also
text3d
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dRotationY

3D text y axis rotation angle (radians).

Possible Values
Floating point value.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. This value specifies the increment by which the text is rotated about the y axis.

See Also
text3d
text3dRotationX
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dRotationZ

3D text z axis rotation angle (radians).

Possible Values
Floating point value.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. This value specifies the increment by which the text is rotated about the z axis.

See Also
text3d
text3dRotationX
text3dRotationY
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dSpeedX

3D text x axis rotation speed.

Possible Values
Floating point value.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. This value specifies the revolutions per second about the x axis.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dSpeedY

3D text y axis rotation speed.

Possible Values
Floating point value.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. This value specifies the revolutions per second about the y axis.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dSpeedZ

3D text z axis rotation speed.

Possible Values
Floating point value.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. This value specifies the revolutions per second about the z axis.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dSpinEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

text3dSpinEnable

Enable/Disable spinning 3D text.

Possible Values
TRUE or FALSE.

Remarks

The 3D text displayed by the Visual 3Space Control can be set to rotate automatically. To turn on spinning 3D text, set text3dSpinEnable to TRUE.

See Also
text3d
text3dRotationX
text3dRotationY
text3dRotationZ
text3dEnable
text3dRed
text3dGreen
text3dBlue
text3dSpeedX
text3dSpeedY
text3dSpeedZ
text3dParts
text3dJustification
text3dComplexity
text3dFontName
text3dFontSize

transparencyType

Sets the transparency quality level to use when rendering.

void setTransparencyType(***long*** *nTranType*)

Possible Values

0 - SCREEN_DOOR
1 - ADD
2 - DELAYED_ADD
3 - SORTED_OBJECT_ADD
4 - BLEND
5 - DELAYED_BLEND
6 - SORTED_OBJECT_BLEND

Remarks

The transparency type determines how transparent objects are rendered.

SCREEN_DOOR
Uses stipple patterns for screen-door transparency.

ADD
Uses additive alpha blending.

DELAYED_ADD
Uses additive blending, rendering all transparent objects after opaque ones.

SORTED_OBJECT_ADD
Same as DELAYED_ADD, but sorts transparent objects by distances of bounding boxes from camera.

BLEND
Uses multiplicative alpha blending.

DELAYED_BLEND
Uses multiplicative alpha blending, rendering all transparend objects after opaque ones.

SORTED_OBJECT_BLEND
Same as DELAYED_BLEND, but sorts transparent objects by distances of bounding boxes from camera.

The default transparency type is SCREEN_DOOR.

See Also

setTransparencyType

traverseViewPoints

Sets the current camera to each position in the view point list.

void traverseViewPoints()

Return Value

None

Parameters

None

Remarks

Each point in the view point list is displayed for the duration specified in addViewPoint.

See Also

addViewPoint
firstViewPoint
lastViewPoint
nextViewPoint
priorViewPoint
resetViewPoints

turnOffLight

Sets the target light to inactive.

void turnOffLight(long lLightId)

Return Value

None.

Parameters

lLightId The ID of the light to turn off

Remarks

When inactive (off), the light does not illuminate at all.

See Also
turnOnLight

turnOnLight

Sets the target light to active.

void turnOnLight(long lLightId)

Return Value

None.

Parameters

lLightId The ID of the light to turn on.

See Also
turnOnLight

urlPickEnable

Enables or disables URL Picking

Possible Values

TRUE - enables URL picking.
FALSE - disables URL picking.

Remarks

URL picking is the process of determing if the mouse pointer is moved over a visible piece of geometry that is a WWW anchor. When URL picking is enabled and viewing mode is off, each time the pointer is moved within the viewer window, a pick calculation is performed to determine if the pointer is over a WWW anchor. If it is and the URL name is different from the the current *pick* URL name, then the UpdatePickUrlName event is fired. Note that the *pick* URL name is not the same as the current URL name. If the left mouse button is clicked while the pointer is over a WWW anchor, then the UrlPicked event is fired.

See Also

isUrlPickEnable
setUrlPickEnable
UpdatePickUrlName
UpdateUrlName
UrlPicked
viewingOn
wwwEnable

validateCameraId

Does the target camera id reference a valid camera ?

BOOL validateCameraId(**long** *lCameraId*)

Return Value

True	The camera id references a valid camera.
False	The camera id does not reference a valid camera.

Parameters

lCameraId	Camera ID to validate.

validateLightId

Does the target light id reference a valid light source ?

BOOL validateLightId(**long** *lLightId*)

Return Value

True	The light id references a valid light source.
False	The light id does not reference a valid light source.

Parameters

lLightId	Light ID to validate.

viewAll

View the entire scene graph.

void viewAll()

Return Value

None.

Parameters

None.

Remarks

Positions the current camera to view the entire scene graph.

viewSelection

Bring the selection into view.

void viewSelection()

Return Value

None.

Parameters

None.

Remarks

Call this method to position the current camera so that the selected object fills the viewer's window.

See Also

setSelectionEnable
selectionEnable
setSelectionMethod
setNoneManip
setCenterballManip
setJackManip
setTabboxManip
setTrackballManip
setTransformBoxManip
setReplaceAllManip

viewerType

The current viewer type.

Possible Values

0	Examiner Viewer
1	Fly Viewer
2	Walk Viewer
3	Plane Viewer

See Also
setExaminerViewer
setFlyViewer
setPlaneViewer
setWalkViewer
SelectViewer
viewerType

viewingOn

Turns viewing mode on or off.

Possible Values

TRUE, turn on viewing.
FALSE, turn off viewing.

Remarks

When viewing mode is turned on, events are consumed by the viewer. When viewing is off, events are processed by the viewer's render area. This means events will be sent down to the scene graph for processing (i.e. picking can occur). Turn viewing mode on to enable movement of the camera through the scene. For example, when the current viewer is the examiner viewer and viewing mode is on, holding down the left mouse button and "grabbing" the object will cause the viewer to rotate the camera about the object. It appears as though the object is spinning but the viewer is actually moving the camera. If you turn viewing mode off and enable selection clicking on components of the scene will cause a bounding box to be displayed around the selected component. You could then replace the bounding box with a trackball manipulator by calling setTrackballManip. If you grab and spin the trackball, the selected component will be rotated. In this case the selected component is actually be moved, not the camera.

See Also

isViewing
setViewing
urlPickEnable
selectionEnable

wwwDdeServiceName

The DDE service name published by the control for viewer remote control.

Possible Values

A valid BSTR.

Remarks

If you want the Visual 3Space Control to use the remote control interface to interact with an HTML browser, you must specify a DDE service name.

See Also
wwwEnable
isWWWEnable
setWWWEnable
urlPickEnable

wwwEnable

Enables HTML browser DDE remote control interface.

Possible Values

TRUE, enable remote control.
FALSE, disable remote control.

Remarks

When this property is set to TRUE, the Visual 3Space Control will publish a DDE service name that enables registration with an HTML browser as a helper application for the x-world/x-inventor and x-world/x-vrml MIME types. When the browser receives files of these mime types, the Visual 3Space control will be sent the necessary information (via a DDE conversation) to load the file. In addition, when a WWW Anchor node is picked in the Visual 3Space control's viewer window, the HTML browser will be instructed to fetch the corresponding URL.

See Also

isWWWEnable
setWWWEnable
wwwDdeServiceName
urlPickEnable